

Claims 1-12 stand rejected under 35 USC §102(e) in view of U.S. Patent Publication No. 2002/0085493 by Pekkala et al. This rejection is respectfully traversed. Applicant's arguments submitted May 4, 2005 are incorporated in their entirety herein by reference. The following comments address the additional issues raised in the Final Action.

Applicant strenuously traverses the unreasonable interpretation of the *explicit claim language* by the Examiner, and the tortured interpretation of Pekkala et al. In particular, the Examiner provides an unreasonable interpretation of the claimed "network node" and "data flow interruption request" to synthesize a rejection that has no rational basis.

Each of the independent claims 1 and 7 specify reducing a prescribed data stream in a network node by reducing execution of a prescribed application resource configured for generating the prescribed data stream: each of the features of detecting a depletion of flow control resources, outputting a data flow interruption request based on the detected depletion of flow control resources, and reducing execution of the prescribed application resource that generates the prescribed data stream, *are performed in the same network node*.

As described in the specification, the claimed network node is illustrated in Fig. 2 (see page 4, lines 9-11 and 19-21) and includes a network interface (e.g., HCA 120), a memory controller 130, system memory 48, and a processor 110 that executes application resources 140a and 140b. The specification explicitly describes that a Host Channel Adapter (HCA) (e.g., 12 of Fig. 1 or 120 of Fig. 2) provides an interface connection to the InfiniBand™ network (e.g., 10 of Fig. 1) (see, e.g., page 2, lines 5-9, page 4, lines 27-31).

Hence, the term "network node" refers to an apparatus that is connected to a network link such as an InfiniBand™ network link. This interpretation also is consistent with the InfiniBand™ Architecture Specification: attached as Exhibit A are pages 41, 44, and 175-176 of the InfiniBand™ Architecture Specification Release 1.0 in effect as of the filing date of the subject application. As shown on page 44 of the InfiniBand™ Architecture Specification, a "Link" is defined as "[a] full duplex transmission path between *any two network fabric elements* [i.e., network nodes], such as Channel Adapters or Switches"; page 41 defines "Channel Adapter" as "[d]evice that *terminates a link* and executes transport-level functions. One of Host

Channel Adapter or Target Channel Adapter.” Further, page 175 of the InfiniBand™ Architecture Specification explicitly states that “[t]he transmitter is the *node* sourcing data packets.”

Therefore, any apparatus that includes an InfiniBand™ link is not a “network node”, as claimed.

The specification also explicitly states that “the network interface (i.e., the HCA 120) detects a *depletion of flow control credits* for an identified virtual lane below a prescribed threshold. The network interface 120 in response outputs in step 210 *a data flow interruption request to the memory controller 130* for the identified virtual lane” (page 5, lines 16-18). Hence, the specification not only specifies that the data flow interruption request is output to the memory controller 130 within the same network node, but also distinguishes between *flow control credits* and the *data flow interruption request*.

The specification also explicitly states that “[t]he memory controller 130, in response to receiving the data flow interruption request, *restricts in step 220 access by the processor 110 to system memory* for the identified virtual lane (service level)” (page 5, lines 19-21); and that “[t]he processor 110, in response to detecting in step 230 the unavailability of the system memory 48 for the identified virtual lane (or the identified service level), *halts in step 240 the execution of the application resources* utilizing the identified virtual lane” (page 5, lines 22-24).

Hence, the specification describes (and the claims require) that the data flow interruption request is output from the network interface to a destination *within the same network node*, and that the processor *within the same network node* reduces the prescribed data stream by reducing execution of the prescribed application resource generating the prescribed data stream; further, the specification requires that the claimed data flow interruption request is *distinct from* existing flow control frames. Any other interpretation would be inconsistent with the interpretation that those skilled in the art would reach, and hence would be unreasonable. Cf. In re Cortright, 49 USPQ2d 1464, 1468 (Fed. Cir. 1999). In fact, “claims are not to be read in a vacuum, and limitations therein are to be interpreted in light of the specification in giving them their ‘broadest

reasonable interpretation.” MPEP § 2111.01 at 2100-37 (Rev. 1, Feb. 2000) (quoting In re Marosi, 218 USPQ 289, 292 (Fed. Cir. 1983)(emphasis in original)).

The Examiner presents an unreasonable interpretation of the claims by asserting on page 2 of the Final Action that “Pekkala does teach the feature of outputting by the network interface in the network node a data flow interruption request as shown in paragraphs 0084, 0092-0094, 0112 [*after detecting that there is no buffers [sic] are available to receive the data packet, issuing and providing a notification/flow control packet (i.e., outputting a data flow interruption request).*]

This assertion that the claimed “data flow interruption *request*” reads on a “flow control *packet*” is unfounded, inconsistent with the specification and claims, and entirely inconsistent with the interpretation of the term “flow control packet” by one skilled in the art. As shown on page 175 of the InfiniBand™ Architecture Specification, “link level flow control [is used] to prevent the loss of packets due to buffer overflow by the receiver *at each end of a link.*” Further, page 175 of the InfiniBand™ Architecture Specification explicitly states that “[t]he transmitter is the *node* sourcing data packets.”

Hence, it is notoriously well known in the art that flow control packets are used to control data flow between link partners at *each end of the link*.

In fact, Pekkala et al. also uses flow control packets to control data flow between link partners (i.e., network nodes) at *each end of the link*: Fig. 1 of Pekkala et al. illustrates a network architecture having a switch 106, hosts 102 (each having an HCA 104) that are connected by InfiniBand™ serial links 132 (see paragraph 44): Fig. 6 of Pekkala et al. describes an improved switch 106 that includes a transaction switch 602, and InfiniBand™ ports 608 (paragraphs 71, 75). These same InfiniBand™ ports 608 are illustrated in Fig. 7 as connected to InfiniBand™ links 132 for connection with a link partner 752 (paragraphs 82-86).

Pekkala et al. describes with respect to Fig. 7 in paragraph 83 that “[t]he port 608 [of the switch 700] comprises an IB transmitter 724 that transmits IB packets, *such as data packets 300 and flow control packets 500, across one half of the full-duplex ID link 132 to a receiver 702 in the link partner 752.*”

Hence, Pekkala et al.: simply describes that the flow control logic 706 "stops" the transmitter 704 in response to receiving the flow control packets 500 *from the port 608 of the switch via the InfiniBand link 132*:

The link partner 752 [of Fig. 7] also includes flow control logic 706 coupled to the receiver 702 and transmitter 704. The link partner 752 flow control logic 706 receives flow control packets 500 from the link partner 752 receiver 702 and provides flow control packets 500 to the link partner 752 transmitter 704. Among other things, the link partner 752 flow control logic 706 responds to flow control packets 500 **received from the port 608 advertising zero credits, and responsively stops the link partner 752 transmitter 704 from transmitting IB data packets 300 to the port 608.**

(Paragraph 84, lines 1-10 of Pekkala et al.).

The independent claims, however, specify outputting *a data flow interruption request*, which is distinct from a conventional flow control *packet* because the flow control *packet must be sent via the network link*. In contrast, the claimed data flow interruption request is kept *within the network node*.

In fact, the Final Action admits in paragraph 5 on page 3 that Pekkala sends *packets to the link partner*.

Hence, the rejection should be withdrawn because it fails to demonstrate that the applied reference discloses each and every element of the claim. See MPEP 2131. "The identical invention must be shown in as complete detail as is contained in the ... claim." Richardson v. Suzuki Motor Co., 868 F.2d 1226, 1236, 9 USPQ2d 1913, 1920 (Fed. Cir. 1989). "Anticipation cannot be predicated on teachings in the reference which are vague or based on conjecture." Studiengesellschaft Kohle mbH v. Dart Industries, Inc., 549 F. Supp. 716, 216 USPQ 381 (D. Del. 1982), aff'd., 726 F.2d 724, 220 USPQ 841 (Fed. Cir. 1984).

Hence, the rejection of independent claims 1 and 7 should be withdrawn.

Further, there is no disclosure or suggestion whatsoever of any memory controller that *renders unavailable* the system memory resources for the prescribed application resource in response to reception of the data flow interruption request, as specified in claims 3 and 8 (and new claim 13).

For these and other reasons, the §102 rejection should be withdrawn.

In view of the above, it is believed this application is and condition for allowance, and such a Notice is respectfully solicited.

To the extent necessary, Applicant petitions for an extension of time under 37 C.F.R. 1.136. Please charge any shortage in fees due in connection with the filing of this paper, including any missing or insufficient fees under 37 C.F.R. 1.17(a), to Deposit Account No. 50-0687, under Order No. 95-508, and please credit any excess fees to such deposit account.

Respectfully submitted,

Manelli Denison & Selter, PLLC

A handwritten signature in black ink, appearing to read 'L. R. Turkevich', with a long horizontal stroke extending to the right.

Leon R. Turkevich
Registration No. 34,035

Customer No. 20736

Date: September 21, 2005

CA	See <u>Channel Adapter</u> .	1
Channel	The association of two queue pairs for communication.	2
Channel Adapter	Device that terminates a link and executes transport-level functions. One of <u>Host Channel Adapter</u> or <u>Target Channel Adapter</u> .	3
Channel Interface	The presentation of the channel to the <u>Verbs Consumer</u> as implemented through the combination of the <u>Host Channel Adapter</u> , associated firmware, and device driver software.	4
Channel, Reliable Datagram	See <u>Reliable Datagram Channel</u> .	5
CI	See <u>Channel Interface</u> .	6
Client	The active entity in an active/passive communication establishment exchange.	7
CM	See <u>Communication Manager</u> .	8
CME	Chassis Management Entity.	9
Communication Manager	The software, hardware, or combination of the two that supports the communication management mechanisms and protocols.	10
Completion Error	Permanent interface or processing error reported through completion status.	11
Completion Queue	A queue containing one or more Completion Queue Entries. Completion Queues are internal to the Channel Interface, and are not visible to verb consumers.	12
Completion Queue Entry	The <u>Channel Interface</u> -internal representation of a <u>Work Completion</u> .	13
Connection	An association between a pair of entities (e.g., processes) over one or more <u>Channels</u> .	14
Consumer	See <u>Verbs Consumer</u> .	15
CQE	<u>Completion Queue Entry</u> , commonly pronounced "cookie".	16
CRC	Cyclic Redundancy Check.	17
Data Payload	The data, not including any control or header information, carried in one packet.	18
Data Segment	A tuple in a <u>Work Request</u> that specifies a virtually contiguous buffer for <u>Host Channel Adapter</u> access. Each Data Segment consists of a Virtual	19


IPv6	Internet Protocol, version 6	1
IPv6 Address	A 128-bit address constructed in accordance with IETF RFC 2460 for IPv6.	2 3 4
Key	A construct used to limit access to one or more resources, similar to a password. The following keys are defined by the InfiniBand™ Architecture:	5 6 7 8
	<u>Baseboard Management Key</u>	9
	<u>Local Key</u>	10
	<u>Management Key</u>	11
	<u>Queue Key</u>	12
	<u>Partition Key</u>	13
	<u>Remote Key</u>	14
L_Key	See <u>Local Key</u> .	15 16
LID	See <u>Local Identifier</u> .	17 18
LID Mask Control	A per-port value assigned by the <u>Subnet Manager</u> . The value of the LMC specifies the number of <u>Path Bits</u> in the <u>Local Identifier</u> .	19 20 21
 Link	A full duplex transmission path between any two network fabric elements, such as <u>Channel Adapters</u> or <u>Switches</u> .	22 23 24
LMC	See <u>LID Mask Control</u> .	25 26
Local Identifier	An address assigned to a port by the <u>Subnet Manager</u> , unique within the subnet, used for directing packets within the subnet. The Source and Destination LIDs are present in the <u>Local Route Header</u> . A Local Identifier is formed by the sum of the <u>Base LID</u> and the value of the <u>Path Bits</u> .	27 28 29 30
Local Key	An opaque object, created by a verb, referring to a <u>Memory Region</u> , used with a Virtual Address to describe authorization for the HCA hardware to access local memory. It may also be used by the HCA hardware to identify the appropriate page tables for use in translating virtual to physical addresses.	31 32 33 34 35
Local Route Header	Routing header present in all InfiniBand™ Architecture packets, used for routing through switches within a subnet.	36 37 38
Local Subnet	The collection of links and <u>Switches</u> that connect the <u>Channel Adapters</u> of a particular subnet.	39 40
LRH	See <u>Local Route Header</u> .	41 42

Table 32 Link Packet

Field	Value
FCCL	0x21B

Generated FCCRC: 0xF9C9

Table 33 Link Packet Byte Stream

Byte	Value
0	0x01
1	0x0D
2	0x52
3	0x1B
4	0xF9
5	0xC9

7.9 FLOW CONTROL

7.9.1 INTRODUCTION

→ This section describes the link level flow control mechanism utilized by IBA to prevent the loss of packets due to buffer overflow by the receiver at each end of a link. This mechanism does not describe end to end flow control such as might be utilized to prevent transmission of messages during periods when receive buffers are not posted.

→ Throughout this section, the terms “transmitter” and “receiver” are utilized to describe each end of a given link. The transmitter is the node sourcing data packets. The receiver is the consumer of the data packets. Each end of the link has a transmitter and a receiver.

IBA utilizes an “absolute” credit based flow control scheme. Unlike many traditional flow control schemes which provide incremental updates that are added to the transmitters available buffer pool, IBA receivers provide a “credit limit”. A credit limit is an indication of the total amount of data that the transmitter has been authorized to send since link initialization.

Errors in transmission, in data packets, or in the exchange of flow control information can result in inconsistencies in the flow control state perceived by the transmitter and receiver. The IBA flow control mechanism provides

for recovery from this condition. The transmitter periodically sends an indication of the total amount of data that it has sent since link initialization. The receiver uses this data to re-synchronize the state between the receiver and transmitter.

7.9.2 FLOW CONTROL BLOCKS

The term “flow control block”, or simply “block” indicates a quantity of data in a data packet. This quantity is defined to be the size of the data packet in bytes (every byte between the local route header and the variant CRC, inclusive) divided by 64 bytes, and rounded up to the next integral value.

7.9.3 RELATIONSHIP TO VIRTUAL LANES

The flow control algorithm defined in this chapter is applied to each virtual lane independently, except for virtual lane 15 which is not subject to link level flow control.

7.9.4 FLOW CONTROL PACKET

Figure 60 Flow Control Packet Format

Flow Control Packet - general format				
bits bytes	31-24	23-16	15-8	7-0
0-3	Op	FCTBS	VL	FCCL
4-5	LPCRC			

C7-53: Flow control packets shall be sent for each VL except VL15 upon entering the LinkInitialize state. When in the PortStates LinkInitialize, LinkArm or LinkActive, a flow control packet for a given virtual lane shall be transmitted prior to the passing of 65,536 symbol times since the last time a flow control packet for the given virtual lane was transmitted.

C7-54: Flow control packets shall use the format specified in [Figure 60 Flow Control Packet Format on page 176](#).

A symbol time is defined as the time required to transmit an eight bit data quantity onto the link. Flow control packets may be transmitted as often as necessary to return credits and enable efficient utilization of the link. See [Section 7.6.4, “Buffering and Flow Control For Data VLs,” on page 149](#) for additional information.

7.9.4.1 FLOW CONTROL PACKET FIELDS

7.9.4.1.1 OPERAND (OP) - 4 BITS

The flow control packet is a link packet with one of two Op (operand) values: An operand of 0x0 indicates a normal flow control packet. An operand value of 0x1 indicates a flow control init packet.